Distributed Spatiotemporal Motion Planning for Spacecraft Swarms in Cluttered Environments

Saptarshi Bandyopadhyay[*], Francesca Baldini[†], Rebecca Foust[‡], Soon-Jo Chung[§], Amir Rahmani[¶], Jean-Pierre de la Croix[‖], Fred Y. Hadaegh[**]

This paper focuses on trajectory planning for spacecraft swarms in cluttered environments, like debris fields or the asteroid belt. Our objective is to reconfigure the spacecraft swarm to a desired formation in a distributed manner while minimizing fuel and avoiding collisions among themselves and with the obstacles. In our prior work we proposed a novel distributed guidance algorithm for spacecraft swarms in static environments.[1] In this paper, we present the Multi-Agent Moving-Obstacles Spherical Expansion and Sequential Convex Programming (MAMO SE–SCP) algorithm that extends our prior work to include spatiotemporal constraints such as time-varying, moving obstacles and desired time-varying terminal positions. In the MAMO SE–SCP algorithm, each agent uses a spherical-expansion-based sampling algorithm to cooperatively explore the time-varying environment, a distributed assignment algorithm to agree on the terminal position for each agent, and a sequential-convex-programming-based optimization step to compute the locally-optimal trajectories from the current location to the assigned time-varying terminal position while avoiding collision with other agent and the moving obstacles. Simulations results demonstrate that the proposed distributed algorithm can be used by a spacecraft swarm to achieve a time-varying, desired formation around an object of interest in a dynamic environment with many moving and tumbling obstacles.

I. Introduction

Trajectory planning for multi-spacecraft formations and swarms, composed of hundreds to thousands of spacecraft, has been a major area of research over the past decade.[2–7] Although there have been significant advances in the development of swarm guidance algorithms for cooperative spacecraft, they cannot be directly applied to handle static and time-varying uncooperative obstacles. In this paper, we present a novel guidance algorithm for spacecraft swarms in an active environment cluttered with many time-varying, moving obstacles, like a debris field or the asteroid belt, and desired time-varying terminal positions. The objective of this algorithm is to reconfigure the swarm to a desired time-varying formation in a distributed manner while minimizing fuel and satisfying spatiotemporal constraints like avoiding collisions among themselves and with the obstacles.

In our prior work, we introduced the Multi-Agent Spherical Expansion and Sequential Convex Programming (Multi-Agent SE–SCP) algorithm for distributed trajectory planning for spacecraft swarms in static environments.[1] In the first step of the Multi-Agent SE–SCP algorithm, the agents use a spherical-expansion-based sampling algorithm to cooperatively explore the workspace and map the asteroid in a collaborative

[*]Robotics Technologist, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, 91109, USA; Saptarshi.Bandyopadhyay@jpl.nasa.gov

[†]Graduate Student, Department of Aerospace, California Institute of Technology, Pasadena, California, 91125, USA; fbaldini@caltech.edu

[‡]Graduate Student, Department of Aerospace Engineering, University of Illinois at Urbana-Champaign, Urbana, Illinois, 61801, USA; foust3@illinois.edu

[§]Associate Professor, Department of Aerospace, California Institute of Technology, Pasadena, California, 91125, USA; sjchung@caltech.edu Senior Member, AIAA.

[¶]Robotics Systems Engineer, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, 91109, USA; Amir.Rahmani@jpl.nasa.gov

[‖]Robotics Systems Engineer, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, 91109, USA; Jean-Pierre.de.la.Croix@jpl.nasa.gov

[**]Senior Research Scientist and Technical Fellow, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, 91109, USA; fred.y.hadaegh@jpl.nasa.gov. Fellow, AIAA.

manner. During the spherical expansion step, each agent stores the position of randomly generated nodes in the free space (the space that is free from obstacles) and the radius of the largest sphere that does not intersect with any obstacle. The agents exchange the positions of the nodes and their radii with their neighboring agents to generate a global view of the workspace while each agent has only explored a much smaller region. This step terminates when all the target positions are strongly connected in the global network of nodes.

Using a distributed auction algorithm[4] the agents converge on an optimal assignment of the target locations in the static desired formation. The agents use their global network of nodes to approximately determine their distance to each of the target locations. During the second step of the Multi-Agent SE–SCP algorithm, each agent generates a locally fuel-optimal trajectory from its current location to its target position using a sequence of convex optimization problems. As the agent moves along this trajectory, it detects the position and velocity of neighboring agents within its vicinity, and uses model predictive control to update its trajectory to avoid colliding with these agents. Thus the swarm achieves the desired formation in a distributed manner while avoiding collisions.

In this paper, we extend the Multi-Agent SE–SCP algorithm for spatiotemporal motion planning of spacecraft swarms in a distributed manner. The objective of the spacecraft swarm is to achieve a desired time-varying formation around a tumbling asteroid, while avoiding collisions among themselves and with the multiple moving obstacles in the active, dynamic environment. The main challenges that arise due to these spatiotemporal constraints, which cannot be addressed by the Multi-Agent SE–SCP algorithm, are:

- The spacecraft swarm need to explore the entire workspace in a cooperative manner because it might be impossible or highly inefficient to explore the entire workspace alone, especially due to its dynamic time-varying nature. Moreover, the spacecraft need to actively keep track of the desired time-varying terminal positions, whose motion is not known a-priori.

- While traveling to their assigned time-varying terminal position, each spacecraft needs to avoid collisions with other spacecraft and the multiple moving obstacles. Moreover, we assume that maximum distance that the terminal position can shift in a time step is less than the maximum distance that the spacecraft can travel in a time step.

In this paper, we address these challenges using the Multi-Agent Moving-Obstacles Spherical Expansion and Sequential Convex Programming (MAMO SE–SCP) algorithm. Our proposed algorithm is computationally efficient, therefore it can be implemented onboard resource-constrained spacecraft. Simulations results demonstrate the effectiveness of the proposed distributed algorithm for guidance of spacecraft swarms.

II. Problem Statement

Let $\mathcal{X} \subset \mathbb{R}^3$ represent the 3D workspace in Local-Vertical-Local-Horizontal (LVLH) frame, as shown in Fig. 1. The workspace $\mathcal{X}$ contains all the initial and terminal positions, moving obstacles, and represents the volume deemed sufficient to find a path for all agents. Let $\mathcal{X}_{\mathrm{obs},k} \subset \mathcal{X}$ represent the time-varying and stationary obstacles in this workspace at the k^{th} time step. Note that the time index is denoted by the subscript. The region where the swarm can maneuver freely is given by $\mathcal{X}_{\mathrm{free},k} = \mathcal{X} / \mathcal{X}_{\mathrm{obs},k}$.

Let N agents belong to this swarm. The initial positions of these N agents are given by $X^i_{\mathrm{init}} \in \mathcal{X}$ for all $i \in \{1, \ldots, N\}$, which are assumed to be stationary. Note that the agent index is denoted by the superscript. Similarly, the N time-varying terminal positions are given by $X^j_{\mathrm{goal},k} \in \mathcal{X}$ for all $j \in \{1, \ldots, N\}$. The actual assignment of the agents to terminal positions will be performed later because the cost-to-go for each agent cannot be calculated beforehand on account of the obstacles. We assume that $\mathcal{X}_{\mathrm{obs},k} \subset \mathcal{X}$ and $X^j_{\mathrm{goal},k} \in \mathcal{X}$ for all $j \in \{1, \ldots, N\}$ are known to each agent at each time step.

To avoid inter-agent collisions, each agent must maintain at least r_{col} distance with every other agent in the swarm. Moreover, let $r_{\max}$ represent the maximum distance that any agent can travel in any time step. We assume that $r_{\max}$ is larger than the maximum distance that any moving obstacle or the time-varying terminal positions can travel in any time step. Moreover, we assume that the initial and final positions satisfy this collision avoidance constraints, i.e.,

$$\left\| X^i_{\mathrm{init}} - X^\ell_{\mathrm{init}} \right\|_2 \geq 2\left(r_{\mathrm{col}} + r_{\max}\right), \qquad \forall i, \ell \in \{1, \ldots, N\}, i \neq \ell \tag{1}$$

$$\left\| X^j_{\mathrm{goal},k} - X^\ell_{\mathrm{goal},k} \right\|_2 \geq 2\left(r_{\mathrm{col}} + r_{\max}\right), \qquad \forall j, \ell \in \{1, \ldots, N\}, j \neq \ell, \quad \forall k \in \mathbb{N} \tag{2}$$

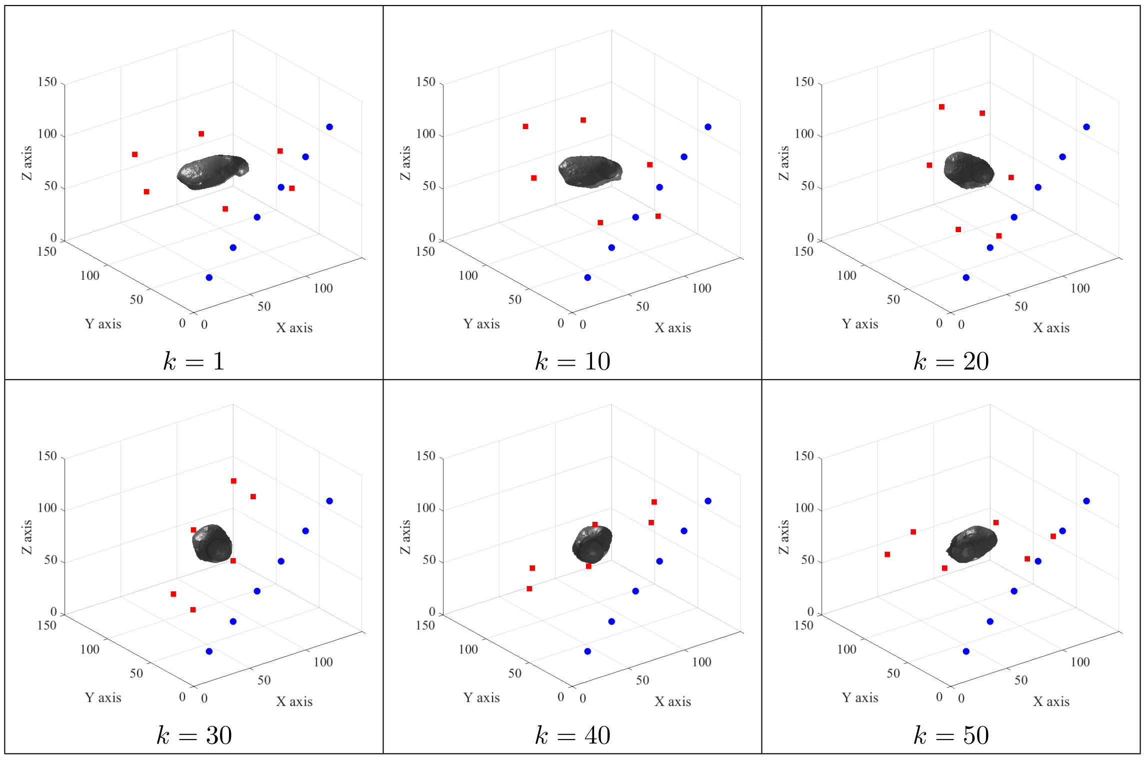

Figure 1: The 3D workspace $\mathcal{X}$, the time-varying obstacles $\mathcal{X}_{\text{obs},k}$, the initial positions $X^i_{\text{init}}, \forall i \in \{1, \dots, N\}$ (in blue), and the time-varying terminal positions $X^j_{\text{goal},k}, \forall j \in \{1, \dots, N\}$ (in red) are shown for $N = 6$ agents and multiple time steps.

The objective of the Multi-Agent Moving-Obstacles Spherical Expansion and Sequential Convex Programming (MAMO SE–SCP) algorithm is to ensure that all the N agents reach the N time-varying terminal positions while avoiding collisions with the time-varying, moving obstacles and among themselves.

III. Multi-Agent Moving-Obstacles Spherical Expansion and Sequential Convex Programming (MAMO SE–SCP) Algorithm

The MAMO SE–SCP algorithm's pseudocode for one of the agents is presented in Algorithm 1. During the *Initialization step*, the necessary data structures are created and initialized. Then the *Spherical Expansion step* and the *Sequential Convex Programming step* are executed iteratively until the agent reaches its terminal position.

Algorithm 1: *Initialization step* of the MAMO SE–SCP Algorithm for the i^{th} agent at the $k = 1$ time step

```
1: $r^i_{\text{init},k} \leftarrow \texttt{MinDistObs}(X^i_{\text{init}}, \mathcal{X}_{\text{obs},k})$
2: $\mathcal{V}^i_k \leftarrow \{X^i_{\text{init}}[r^i_{\text{init},k}]\}$
3: $\mathcal{E}^i_k \leftarrow \emptyset$, $F^i_{\text{reached}} \leftarrow 0$, $F^i_{\text{connected}} \leftarrow 0$, $X^i_{\text{term}} \leftarrow \emptyset$
```

III.A. Initialization Step

The i^{th} agent's MAMO SE–SCP algorithm intends to generates a time-varying directed graph $\mathcal{G}^i_k = (\mathcal{V}^i_k, \mathcal{E}^i_k)$ in the time-varying safe region $\mathcal{X}_{\text{free},k}$. Each node in the set of nodes $\mathcal{V}^i$ stores the position of the node and the minimum distance of that node from any obstacle (both in $\mathcal{X}_{\text{obs},k}$ and other agents). For the node X^i_{init}, the minimum distance $r^i_{\text{init},k}$ from the obstacle $\mathcal{X}_{\text{obs},k}$ is obtained using the function $\texttt{MinDistObs}(X_{\text{init}}, \mathcal{X}_{\text{obs},k})$, which takes in the position of the node and the obstacles in the workspace and returns the radius of the largest sphere centered on that node which does not intersect with any obstacle. Then the set of nodes $\mathcal{V}_k^i$ is initialized with the nodes $X_{\text{init}}^i[r_{\text{init},k}^i]$.

Each element in the set of edges $\mathcal{E}_k^i$ stores the edge's starting and ending nodes and the cost of traversing that edge. The set of edges $\mathcal{E}_k^i$ is initialized with the empty set. The flag F_{reached}^i, which denotes if the i^{th} agent has reached its terminal position, is set to zero. The flag $F_{\text{connected}}^i$, which denotes the number of terminal positions that the i^{th} agent is connected to, is also set to zero. The assigned terminal position X_{term}^i of the i^{th} agent is set to an empty set.

III.B. Spherical Expansion Step

During this step, the workspace is explored using the sampling technique shown in lines 5–29 in Algorithm 1. The objective of this step is to populate the graph $\mathcal{G}_k^i = (\mathcal{V}_k^i, \mathcal{E}_k^i)$ in order to find paths from X_{init}^i to $X_{\text{goal},k}^j$ for all $j \in \{1, \ldots, N\}$.

Algorithm 1: ***Spherical Expansion step*** of the MAMO SE–SCP Algorithm for the i^{th} agent at the k^{th} time step

```
 4: if $\sum_{\ell=1}^{N} F_{\text{reached}}^{\ell} \neq N$ then
 5:     $Y_k^\ell, X_{\text{new}}^\ell, F_{\text{connected}}^\ell, F_{\text{reached}}^\ell, \forall \ell \in \{1,\ldots,N\} \leftarrow$ AllAgentCommunicate
 6:     $\hat{\mathcal{X}}_{\text{obs},k}^i = \mathcal{X}_{\text{obs},k}$
 7:     for $\ell = \{1,\ldots,N\}/\{i\}$ do
 8:         $\hat{\mathcal{X}}_{\text{obs},k}^i = \hat{\mathcal{X}}_{\text{obs},k}^i \cup$ GenerateSphere$(Y_k^\ell, r_{\text{col}} + r_{\max})$
 9:         $\mathcal{V}_k^i \leftarrow \mathcal{V}_k^i \cup \{X_{\text{new}}^\ell[0]\} \cup \{X_{\text{goal},k}^j[0]\}$
10:     end for
11:     $\mathcal{V}_{\text{new}}^i \leftarrow \emptyset$
12:     for all $X_{\text{v}}[r_{\text{v},k}] \in \mathcal{V}_k^i$ do
13:         $r_{\text{v},k} \leftarrow$ MinDistObs$(X_{\text{v}}, \hat{\mathcal{X}}_{\text{obs},k}^i)$
14:         $\mathcal{V}_{\text{new}}^i \leftarrow \mathcal{V}_{\text{new}}^i \cup \{X_{\text{v}}[r_{\text{v},k}]\}$
15:     end for
16:     $\mathcal{V}_k^i \leftarrow \mathcal{V}_{\text{new}}^i$
17:     $X_{\text{rand}} \leftarrow$ GenerateSample
18:     $X_{\text{nearest}} \leftarrow$ NearestNode$(\mathcal{V}^i, X_{\text{rand}})$
19:     $X_{\text{new}}^i \leftarrow$ Steer$(X_{\text{rand}}, X_{\text{nearest}})$
20:     $r_{\text{new},k}^i \leftarrow$ MinDistObs$(X_{\text{new}}^i, \hat{\mathcal{X}}_{\text{obs},k}^i)$
21:     $\mathcal{V}_k^i \leftarrow \mathcal{V}_k^i \cup \{X_{\text{new}}^i[r_{\text{new},k}^i]\}$
22:     $\mathcal{E}_k^i \leftarrow \emptyset$
23:     for all $X_{\text{v}}[r_{\text{v},k}], X_{\text{w}}[r_{\text{w},k}] \in \mathcal{V}_k^i$ and $X_{\text{v}} \neq X_{\text{w}}$ do
24:         if $\|X_{\text{v}} - X_{\text{w}}\|_2 \leq r_{\text{v},k} + r_{\text{w},k}$ then
25:             $c_{\text{v,w}} \leftarrow$ EdgeCost$(X_{\text{v}}, X_{\text{w}})$
26:             $c_{\text{w,v}} \leftarrow$ EdgeCost$(X_{\text{w}}, X_{\text{v}})$
27:             $\mathcal{E}_k^i \leftarrow \mathcal{E}_k^i \cup \{\overrightarrow{X_{\text{v}} X_{\text{w}}}[c_{\text{v,w}}]\} \cup \{\overrightarrow{X_{\text{w}} X_{\text{v}}}[c_{\text{w,v}}]\}$
28:         end if
29:     end for
30: end if
```

Let $Y_k^\ell \in \mathcal{X}$ for all $\ell \in \{1, \ldots, N\}$ represent the current position of each agent at the k^{th} time step. All the agents exchange their position Y_k^ℓ, their new nodes X_{new}^ℓ, and their flags $F_{\text{connected}}^\ell$ and F_{reached}^ℓ using the function `AllAgentCommunicate`, which relies on inter-agent communication and a strongly-connected communication network topology. During the first iteration, no new nodes are communicated. Communicating only the new nodes, as opposed to complete trajectories or other features of the obstacles, allows the agents to collaboratively explore the workspace in a computationally efficient manner under limited bandwidth constraints.

The lines 6–10 create a new obstacle set $\hat{\mathcal{X}}_{\text{obs},k}^i$ where the original obstacle set $\mathcal{X}_{\text{obs},k}$ is augmented with spheres of radius $(r_{\text{col}} + r_{\max})$ centered on the position of all the other agents. Thus $\mathcal{X}_{\text{free},k}^i = \mathcal{X}/\hat{\mathcal{X}}_{\text{obs},k}^i$ represents the region where the i^{th} agent can maneuver freely at the k^{th} time step.

The new nodes from other agents $X_{\text{new}}^\ell, \forall \ell \in \{1, \ldots, N\}$ and the new terminal positions $X_{\text{goal},k}^j, \forall j \in$

$\{1, \ldots, N\}$ are also added to $\mathcal{V}^i$ during lines 6–10. The lines 11–16 are used to update the radius of the nodes in $\mathcal{V}^i$ with the new obstacles set $\hat{\mathcal{X}}^i_{\text{obs},k}$. The process of adding a new node and regerating the edges of the graph $\mathcal{G}^i_k = (\mathcal{V}^i_k, \mathcal{E}^i_k)$ is shown in lines 17–29. It is similar to that of the original Multi-Agent SE–SCP algorithm,[1] hence it is not explained here for brevity.

For the problem setup shown in Fig. 1, multiple iterations of only the Spherical Expansion step are shown in Fig. 2. Note that each agent is able to generate a dense graph within 30 iterations because each agent also uses the nodes generated by other agents.

III.C. Sequential Convex Programming Step

During this step in lines 32–60 in Algorithm 1, each agent first determines its terminal position and then moves towards that terminal position by generating locally optimal trajectories. Since this step is similar to that of the original Multi-Agent SE–SCP algorithm,[1] we only highlight the differences here for brevity.

Algorithm 1: *Sequential Convex Programming step* of the MAMO SE–SCP Algorithm for the i^{th} agent at the k^{th} time step

```
31: if $\sum_{\ell=1}^{N} F^{\ell}_{\text{reached},k-1} \neq N$ then
32:     if $X^i_{\text{term},k} = \emptyset$ then
33:         if $\sum_{\ell=1}^{N} F^{\ell}_{\text{connected}} = N^2$ then
34:             $X^i_{\text{term},k} \leftarrow \texttt{DistributedAssignment}$,
35:             $\mathcal{V}^i_k \leftarrow \mathcal{V}^i_k \cup \{X^i_{\text{term},k}[0]\}$
36:         else
37:             $F^i_{\text{connected}} \leftarrow 0$
38:             for $j = \{1, \ldots, N\}$ do
39:                 $P^{i,j}_k, c_{P^{i,j}_k} \leftarrow \texttt{MinPath}(\mathcal{G}^i_k = (\mathcal{V}^i_k, \mathcal{E}^i_k), X^i_{\text{init}}, X^j_{\text{goal},k})$
40:                 if $c_{P^{i,j}_k} < \infty$ then
41:                     $F^i_{\text{connected}} \leftarrow F^i_{\text{connected}} + 1$
42:                 end if
43:             end for
44:         end if
45:     else
46:         if $X^i_{\text{term},k} = Y^i_k$ then
47:             $F^i_{\text{reached}} \leftarrow 1, \quad \boldsymbol{x}^i \leftarrow \emptyset$
48:         else
49:             $F^i_{\text{reached}} \leftarrow 0$
50:             $P^i, c_{P^i} \leftarrow \texttt{MinPath}(\mathcal{G}^i_k = (\mathcal{V}^i_k, \mathcal{E}^i_k), Y^i_k, X^i_{\text{term},k})$
51:             $(\boldsymbol{x}^i_1, \boldsymbol{u}^i_1, c_{\boldsymbol{x}^i_1}) \leftarrow \texttt{OptimalTraj}(P^i)$
52:             for $\tau = \{1, \ldots, N_{SCP}\}$ do
53:                 $P^i_\tau \leftarrow \texttt{GeneratePath}(\boldsymbol{x}^i_\tau)$
54:                 $(\boldsymbol{x}^i_{\tau+1}, \boldsymbol{u}^i_{\tau+1}, c_{\boldsymbol{x}^i_{\tau+1}}) \leftarrow \texttt{OptimalTraj}(P^i_\tau, \boldsymbol{x}^i_\tau, \boldsymbol{u}^i_\tau)$
55:             end for
56:             $\boldsymbol{x}^i \leftarrow \boldsymbol{x}^i_{N_{SCP}+1}$
57:         end if
58:     end if
59:     $Y^i_{k+1} \leftarrow \texttt{AgentMotion}(\boldsymbol{x}^i)$
60:     $\mathcal{V}^i_k \leftarrow \mathcal{V}^i_k \cup \{Y^i_{k+1}[0]\}$
61: end if
```

If all the agents are connected to all the time-varying terminal positions, then the agent execute a distributed assignment algorithm, like linear programming,[8] auction algorithm,[9] and variable-target-number distributed auction algorithm,[4] to converge on a suitable assignment of terminal positions (line 34). Otherwise the i^{th} agent counts the number of terminal positions it is connected to in lines 37–43. The function $\texttt{MinPath}(\mathcal{G}^i_k = (\mathcal{V}^i_k, \mathcal{E}^i_k), X^i_{\text{init}}, X^j_{\text{goal},k})$ takes in the current graph, the initial and goal positions, and returns the minimum-cost path $P^{i,j}_k$ along with the cost of that path $c_{P^{i,j}_k}$. The path $P^{i,j}_k = \{X_1[r_{1,k}], X_2[r_{2,k}], \ldots, X_{\text{m}}[r_{\text{m},k}]\}$ is a sequence of m nodes with corresponding radii, where $X_1 = X^i_{\text{init}}$ and $X_{\text{m}} = X^j_{\text{goal},k}$. The cost

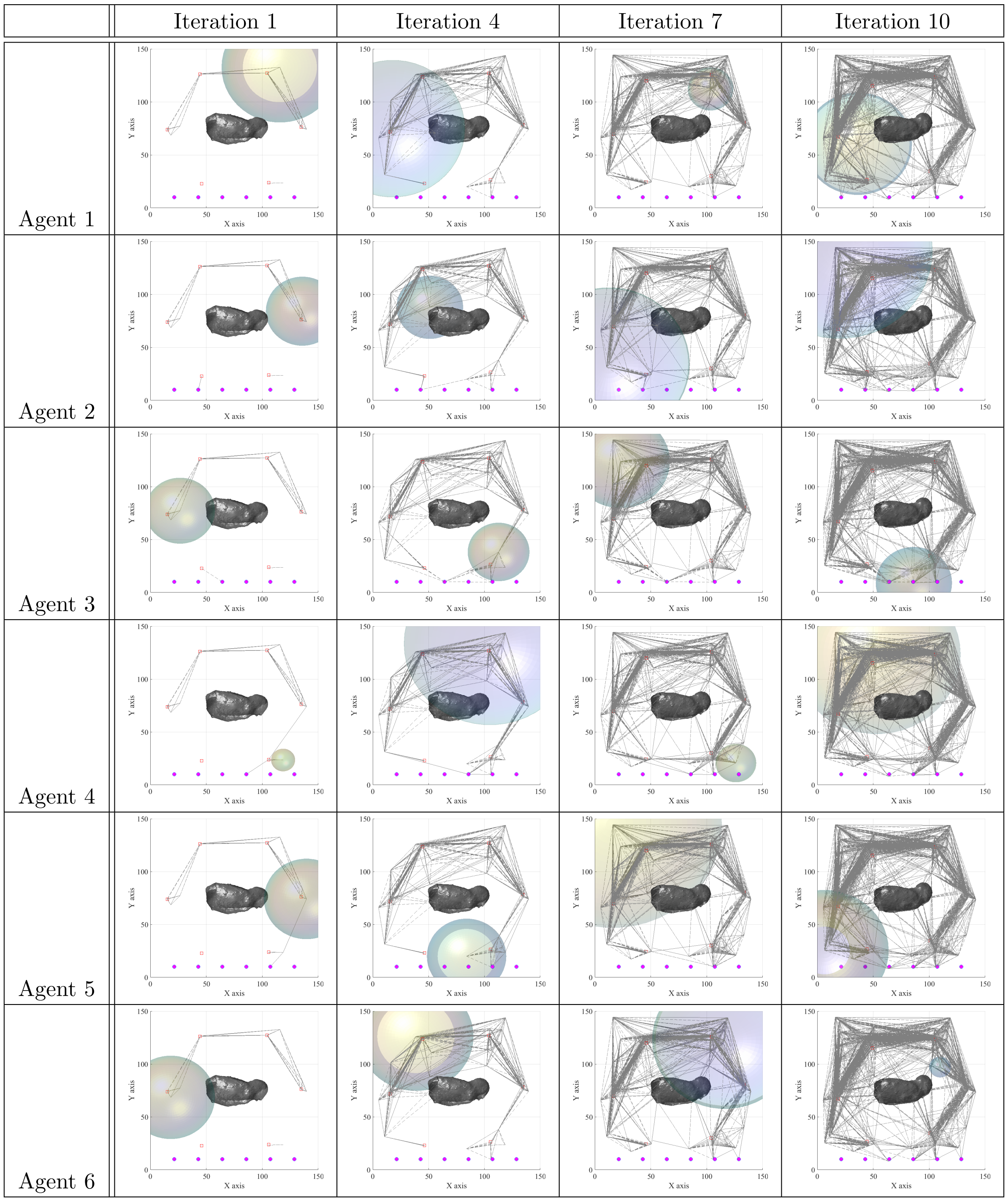

Figure 2: Multiple iterations of the Spherical Expansion step in the MAMO SE–SCP algorithm are shown. All agents (in magenta) are located at their starting positions. Note that the desired time-varying terminal positions matintain a constant relative attitude with respect to the tumbling asteroid.

of the path $c_{P_k^{i,j}}$ is the sum of the edges along that path. Graph search algorithms like Dijkstra's algorithm can be used for this step.

If the terminal position $X^i_{\mathrm{term},k}$ is assigned (line 45), then the path from the current position Y^i_k to the terminal position $X^i_{\mathrm{term},k}$ is found using multiple iterations of the function `OptimalTraj`. This optimization step is similar to that of the original Multi-Agent SE–SCP algorithm.[1] A few steps of the MAMO SE–

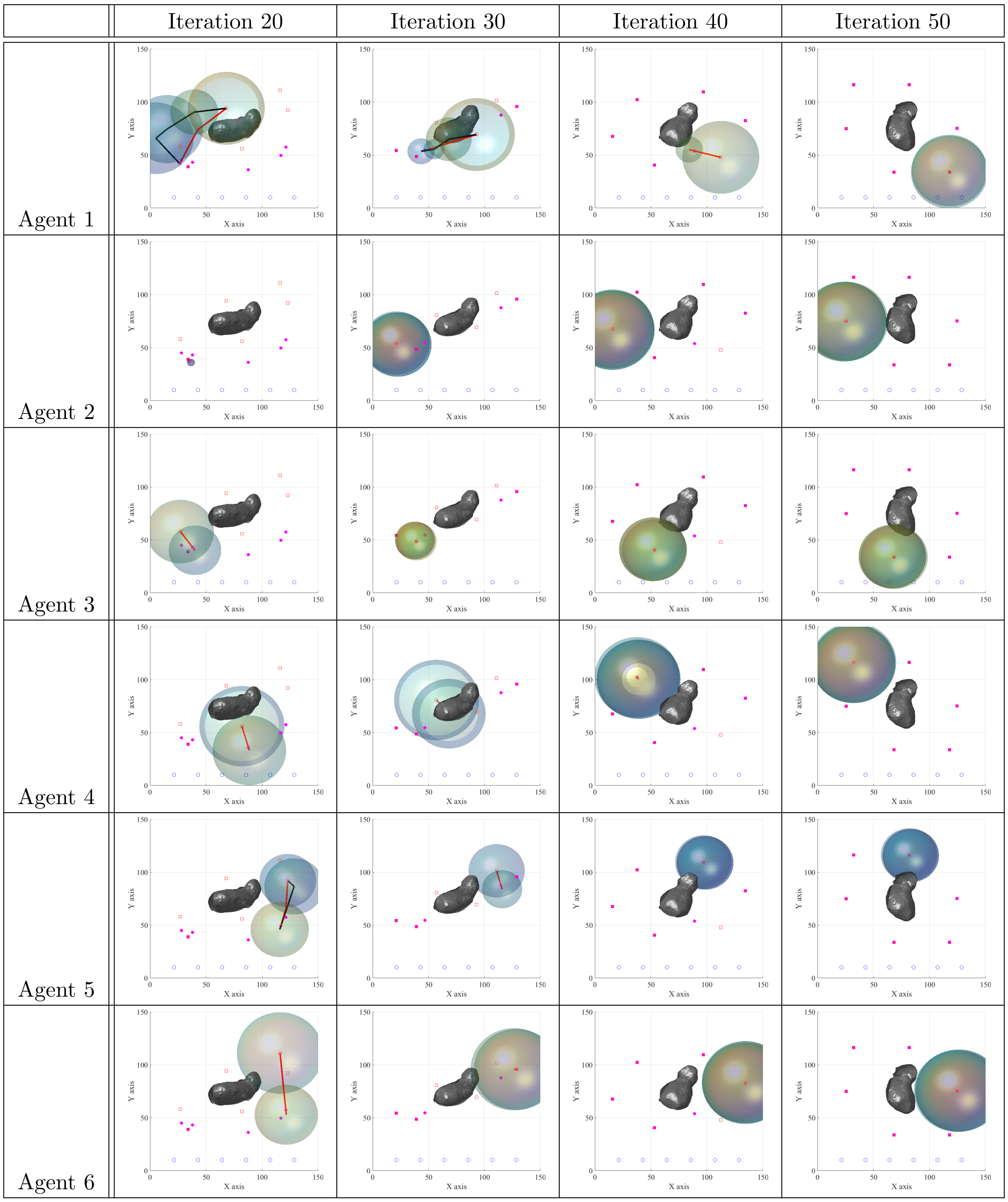

Figure 3: Multiple iterations of the entire MAMO SE–SCP algorithm are shown. All agents (in magenta) are moving to their terminal positions. Note that the agents keep tracking their desired time-varying terminal positions after reaching them.

SCP algorithm are show in Fig. 3. Note that each agent updates its optimal trajectory based on its current location and the location of other agents while avoiding collisions with other agents and the moving obstacles.

IV. Numerical Simulations

In this section, we use the MAMO SE–SCP algorithm to reconfigure a swarm of 48 agents into the desired time-varying formation around a tumbling asteroid shown in Fig. 4. The terminal positions form a spherical formation and maintain a constant attitude around the central tumbling asteroid. A number of moving, tumbling obstacles are also present in this active environment.

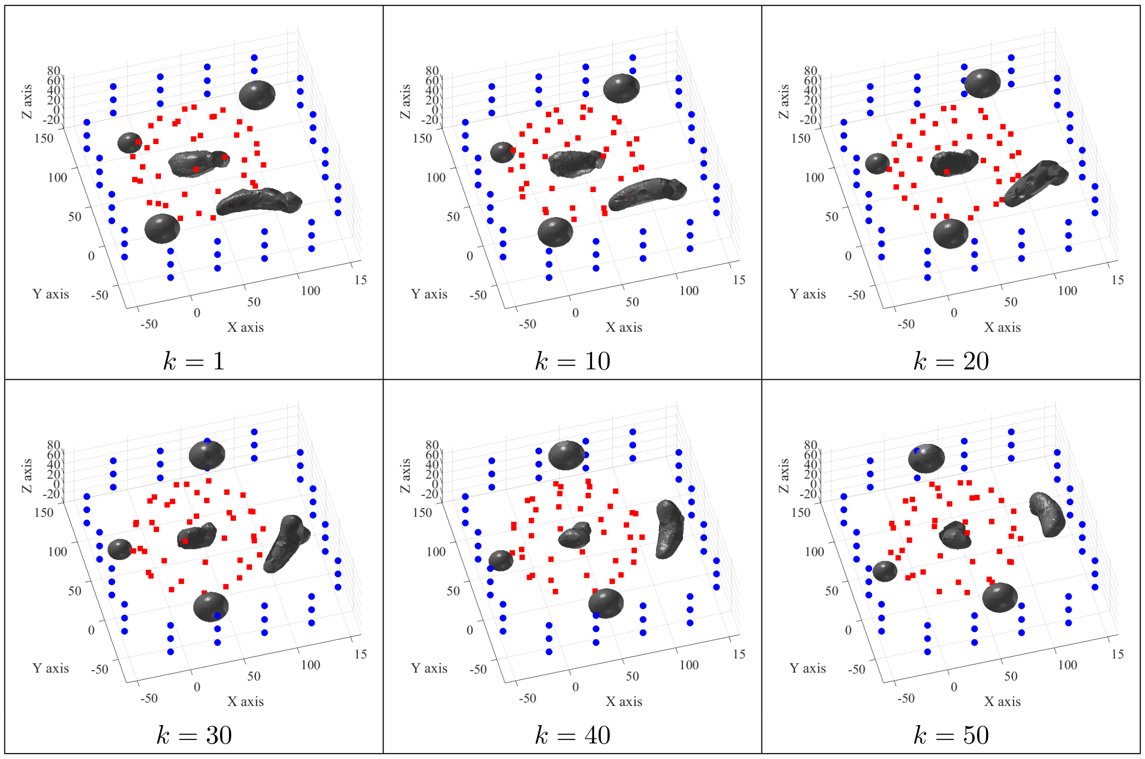

Figure 4: The 3D workspace $\mathcal{X}$, the moving obstacles $\mathcal{X}_{\text{obs},k}$, the initial positions X^i_{init} (in blue), and the time-varying terminal positions $X^j_{\text{goal},k}$ (in red) are shown for $N = 48$ agents. Note that the desired time-varying terminal positions matintain a constant relative attitude with respect to the central tumbling asteroid.

Multiple iterations of the MAMO SE–SCP algorithm for agents 12, 24, 36, and 48 are shown in Fig. 5 and 6. During the first 19 iterations of the Spherical Expansion step, each agent is able to generate a dense graph of the time-varying workspace. During the remaining 30 iterations, each agent travels to its chosen terminal position while avoiding collisions with other agents and the moving obstacles.

V. Conclusions

In this paper, we extend our prior work on motion planning of spacecraft swarm in static cluttered environments[1] to handle spatiotemporal constraints such as time-varying, moving obstacles and desired time-varying terminal positions. The first step in the algorithm is a spherical-expansion-based sampling algorithm to cooperatively explore the time-varying environment and map the moving obstacles in the environment where the agents exchange the positions of the nodes and their radii with their neighboring agents to generate a global view of the workspace while each agent has only explored a much smaller region. After a distributed assignment step, the agent generates a locally fuel-optimal trajectory from its current location to its time-varying target position using a sequence of convex optimization problems. As the agent moves along this trajectory, it detects the position of other agents and moving obstacles, and updates its trajectory to avoid collisions with other agents and the moving obstacles. Thus the swarm achieves the desired time-varying formation in a distributed manner while avoiding collisions. The MAMO SE–SCP algorithm is computationally efficient, therefore it can be implemented onboard resource-constrained spacecraft. Simulations results demonstrate the effectiveness of the proposed distributed algorithm for guidance of spacecraft

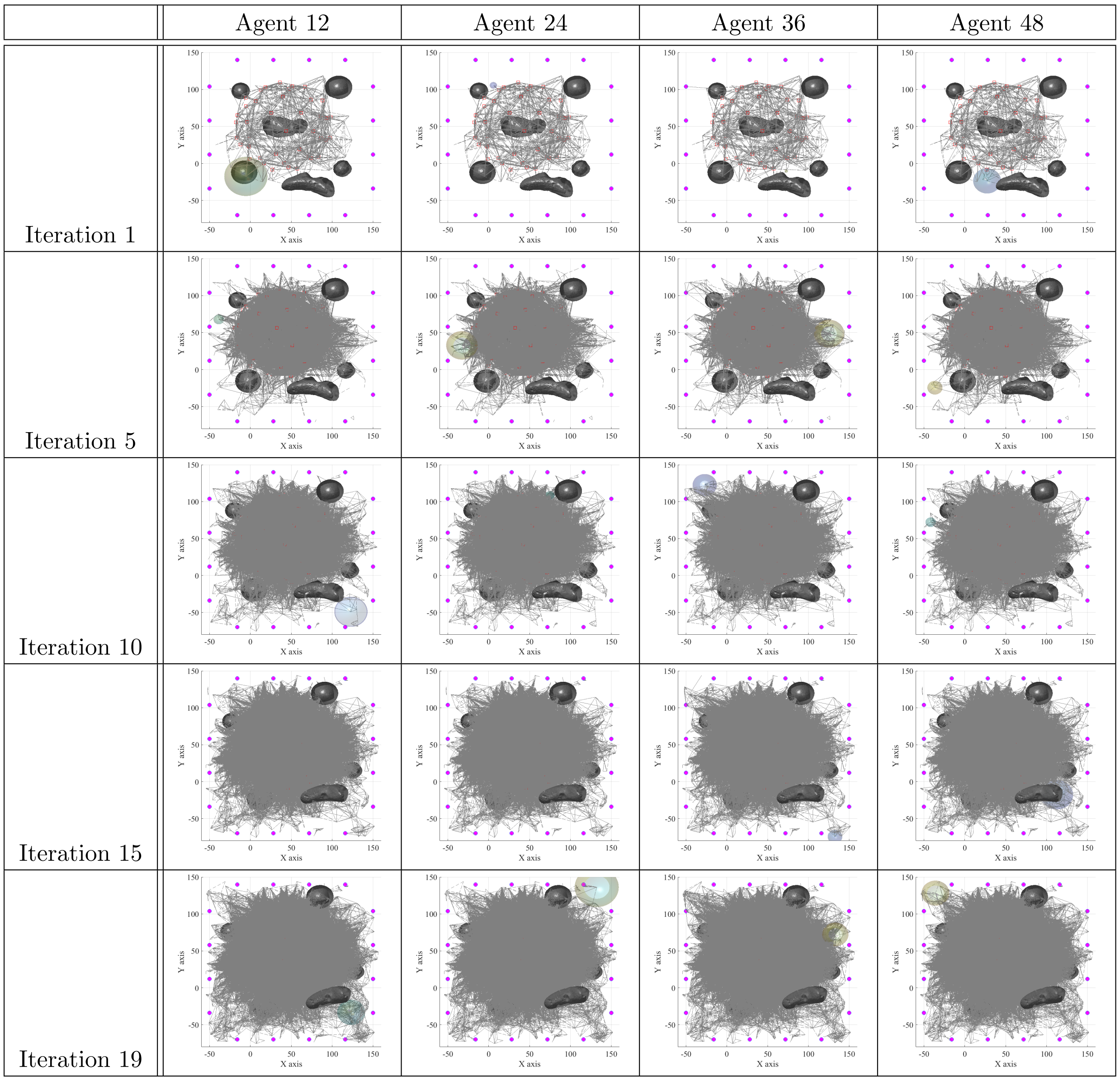

Figure 5: Multiple iterations of the the Spherical Expansion step in the of the MAMO SE–SCP algorithm are shown. All agents (in magenta) are located at their starting positions.

swarms in dynamic environments.

Acknowledgment

This work was supported by the Jet Propulsion Laboratory's Research and Technology Development (R&TD) program. Part of the research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. ©2017 California Institute of Technology. Government sponsorship acknowledged.

References

[1]Bandyopadhyay, S., Baldini, F., Foust, R., Chung, S.-J., Rahmani, A., de la Croix, J.-P., and Hadaegh, F. Y., "Distributed Fast Motion Planning for Spacecraft Swarms in Cluttered Environments using Spherical Expansions and Sequence of Convex

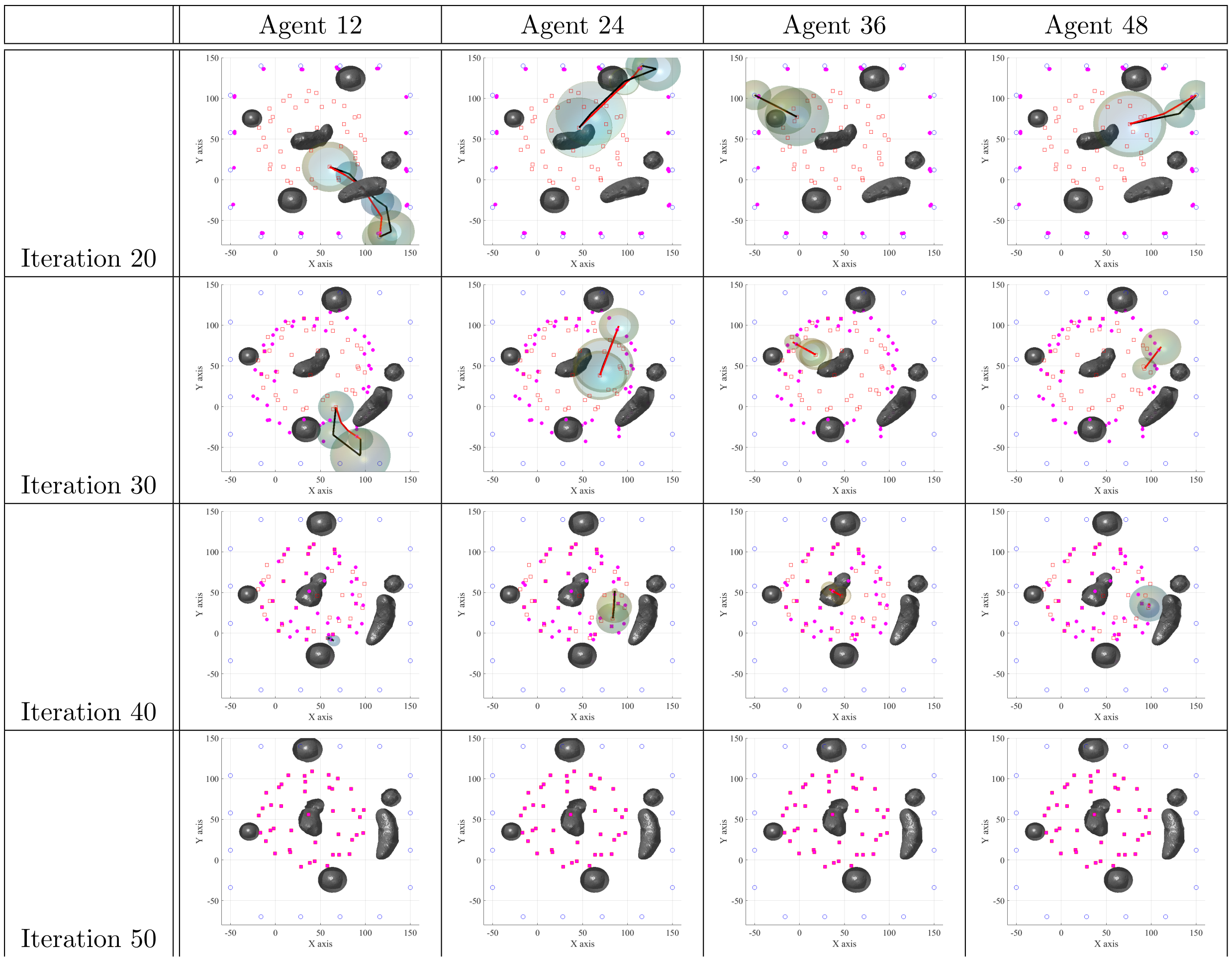

Figure 6: Multiple iterations of the entire MAMO SE–SCP algorithm are shown. The location of the agents are shown using magenta dots.

Optimization Problems," *9th International Workshop on Satellite Constellations and Formation Flying*, Boulder, CO, June 2017.

[2]Bandyopadhyay, S., Foust, R., Subramanian, G. P., Chung, S.-J., and Hadaegh, F. Y., "Review of Formation Flying and Constellation Missions using Nanosatellites," *J. Spacecraft and Rockets*, Vol. 53, No. 3, 2016, pp. 567–578.

[3]Morgan, D., Chung, S.-J., Blackmore, L., Acikmese, B., Bayard, D., and Hadaegh, F. Y., "Swarm-Keeping Strategies for Spacecraft under J2 and Atmospheric Drag Perturbations," *J. Guid. Control Dyn.*, Vol. 35, No. 5, 2012, pp. 1492 – 1506.

[4]Morgan, D., Subramanian, G. P., Chung, S.-J., and Hadaegh, F. Y., "Swarm Assignment and Trajectory Optimization Using Variable-Swarm, Distributed Auction Assignment and Sequential Convex Programming," *Int. J. Robotics Research*, Vol. 35, 2016, pp. 1261–1285.

[5]Bandyopadhyay, S., Chung, S.-J., and Hadaegh, F. Y., "Probabilistic and Distributed Control of a Large-Scale Swarm of Autonomous Agents," *IEEE Trans. Robotics*, 2017, to appear.

[6]Bandyopadhyay, S., Chung, S.-J., and Hadaegh, F. Y., "A Probabilistic Eulerian Approach for Motion Planning of a Large-Scale Swarm of Robots," *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Daejeon, South Korea, Oct. 2016.

[7]Morgan, D., Subramanian, G. P., Bandyopadhyay, S., Chung, S.-J., and Hadaegh, F. Y., "Probabilistic guidance of distributed systems using sequential convex programming," *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Chicago, IL, Sept. 2014, pp. 3850–3857.

[8]Richert, D. and Cortés, J., "Robust distributed linear programming," *IEEE Trans. Autom. Control*, Vol. 60, No. 10, 2015, pp. 2567–2582.

[9]Zavlanos, M. M., Spesivtsev, L., and Pappas, G. J., "A Distributed Auction Algorithm for the Assignment Problem," *IEEE Conf. Decision Control*, Cancun, Mexico, Dec. 2008, pp. 1212–1217.